January/February 1987 Edition

RE RUN

RUN Programs on Disk

*For the C-64 and C-128**

Inside:
Bonus Programs from RUN SPECIAL ISSUE!

*128 mode programs included

Introduction

January-February '87 ReRUN

Welcome to our first 1987 edition of ReRUN. This disk contains C-128 and C-64 programs from the January and February issues of *RUN* magazine, plus all of the type-in programs from *RUN*'s third annual Special Issue and *RUN*'s Index of 1986 articles and reviews. Let's begin with a look at the C-128 programs.

Reminder 128 is an application that you can use throughout the year. It's an 80-column appointment book that lets you enter up to 100 dates and events.

RUN Script users will appreciate Patching up RUN Script 128, a program that adds new printing capabilities to that versatile word processor. Now, you can create documents with double-sided pages and columns of text. Also, for RUN Script users, we have the Define Macros program for creating macro tables.

Three applications for handling money and figures are included on this disk. Retir'eze helps you manage your annual expenses by calculating the future effects of present financial moves. Mind Your Mortgage figures out loan payments and repayment schedules on a proposed mortgage. It also gives you a monthly amortization schedule. If you'd like to use your C-128 for fast calculations, check out Add/Calc 128. This little program turns your computer into a printing calculator.

We have three versatile C-128 utilities on this disk, too. February's Mega-Magic program, 128 HI RES DUMP, lets you print out high-resolution screens. Break the 128 Memory Barrier! lets you create a RAM disk by utilizing the RAM expansion module for the C-128. The third utility, The Light Choice, is a menu highlighter that programmers can use to spruce up their menus.

Take a break with Twiddle, a game in which you and your opponent try to outmaneuver each other on a 5 × 5 game board, reminiscent of checkers.

If you're an applications user, you're going to have fun with the seven for the C-64 that appear on this disk. 64 Notepad Update is an enhancement of the 64 Notepad program (*RUN*, Sep-

tember 1986) and is used for saving and loading any notepad window that you've created. You also get a digital time display.

Calendar lets you print out any month of the year. Label Copy 64 and Monthly Labels will take care of all your label needs, and you can use Envelope Maker if you don't have any envelopes to put your labels on.

The Datafile series is being published in several issues of *RUN* this year. February held the main program, Datafile 3.6, along with two short utilities for fast sorting and using the DOS 5.1 wedge. You need DFPrint in order to print out your records, so be sure to get that program in March's *RUN*.

People on the road are using portable computers, such as the Radio Shack Model 100, to handle their computing needs. We have a Model 100-to-64 terminal program that lets owners of these two computers share files. Take a look at Hook Up to a Portable for details.

Programmers will want to check out several C-64 routines on this disk. January's Mega-Magic program performs numeric sorting. Solving the Split-Word Problem lets programmers implement word wrap in their programs. For your C-64 programs that have menus, take a look at Master Menus.

Educational programs on this disk include Crosswords, Spelling and Math Functions.

As an added bonus, the Index to *RUN*'s 1986 Articles and Reviews is included in a sequential file. Just load this into your word processor and print it out for a quick reference guide to what *RUN* published in 1986.

That's all for this edition. Enjoy the programs.

Margaret Morabito
Technical Manager
RUN *magazine*

How To Load

Loading from Menu

To get started, C-64 users should type LOAD "MENU 64",8 and press the return key. When you get the Ready prompt, the menu is loaded and you should type RUN to see a list of the programs on your disk.

Loading from Keyboard

If you do not wish to use the menu program, follow these instructions.

C-64:

To load a C-64 program written in Basic, type:

LOAD "DISK FILENAME",8

and then press the return key. The drive will whir while the screen prints LOADING and then READY, with a flashing cursor beneath. Type RUN and press the return key. The program will then start running.

To load a C-64 program written in machine language (ML), type:

LOAD "DISK FILENAME",8,1

C-128:

All C-64 programs can be run on the C-128 as long as your computer is in C-64 mode.

All C-128 programs are clearly labeled on the directory page. Your C-128 *must* be in C-128 mode to run these programs.

To load a C-128-mode program, press the F2 key, type the disk filename and then press the return key. When the program has loaded, type RUN.

Making Copies of ReRUN Disks

Many of the programs on your ReRUN disk have routines that require you to have a separate disk onto which the program writes or saves subfiles. In order for you to use these programs, you will first have to make a copy of the original program onto another disk that has enough free space on it to hold these newly written subfiles.

If the program is written in Basic, it is simple to make a copy of the program. Just load the program into your computer following the procedures outlined above, and then save the program back onto a separate disk that has plenty of free space for extra files.

If the program is written in ML, copying is not so simple. You cannot simply load and save an ML program. In this case, you'll need to use a disk-backup utility program, such as the one on your Commodore Test Demo disk.

Directory

Page	Article	Disk Filename	File Type
	C-128 Programs		
		MENU 128	BASIC
1	Reminder 128	REMINDER 128	BASIC
3	Patching Up RUN Script 128	PATCH RS128	BASIC
6	Mega-Magic February	128 HI RES DUMP	BASIC
7	Retir'eze	RETIREZE 128/64	BASIC
11	Twiddle	TWIDDLE 128	BASIC
16	Break the 128 Memory Barrier!	RAM DISK LOADER	BASIC
19	Mind Your Mortgage	LOAN ANALYSIS	BASIC
21	The Light Choice	MENU HIGHLIGHTER	BASIC
24	Add/Calc 128	ADD/CALC 128	BASIC
	C-64 Programs		
		MENU 64	BASIC
26	64 Notepad Update	NOTEPAD 1	BASIC
		NOTEPAD 2	BASIC
		NOTEPAD 3	BASIC
29	RUN Script 128, Part 2	DEFINE MACROS	BASIC
32	Word Wars	CROSSWORDS	BASIC

Page	Article	Disk Filename	File Type
	C-64 Programs		
	From January Resource Center	SPELLING	BASIC
36	Mega-Magic January	SORTER 1	BASIC
		SORTER 2	BASIC
		SORTER 3	BASIC
38	Keeping Up to Date	CALENDAR	BASIC
39	Solving the Split-Word Problem	SPLIT WORD1	BASIC
		SPLIT WORD2	BASIC
43	Hook Up to a Portable	M100 TO C64	M/L
47	Datafile 3.6	DATAFILE 3.6	BASIC
		SORT GENERATOR	BASIC
		INSTALL DOS5.1	BASIC
52	The Functional Computer	MATH FUNCTIONS	BASIC
55	Monthly Labels	MONTHLY LABELS	BASIC
56	Lots of Labels	LABEL COPY 64	BASIC
57	Envelope Maker	ENVELOPE MAKER	BASIC
60	Master Menus	ML MENU LOADER	BASIC
		MENU DEMO	BASIC
	1986 Index of Articles	1986 INDEX*	SEQ

NOTE: Read articles before running these programs!

* This sequential file must be loaded into a word processor.

Reminder 128

By Bob Guerra and Jim Richards

RUN It Right

C-128 (in 80-column mode); disk drive

If you have a lot of important dates to remember, you know how useful a desk calendar or pocket date book can be. Reminder 128 is an electronic desk calendar that improves on the pencil-and-paper versions. It eliminates thumbing through pages by letting you store up to 100 dates and then search for upcoming events by typing in the current date.

Before running Reminder 128, copy the program to a new disk! The first time you boot Reminder 128 and type in the current date, the program automatically creates a relative file called REMFILE for storing your reminders. This file is saved onto your new disk. Once this has been done, you can press any key to access the main screen and search the file. Since the file will be new at this point, the program will say you have no messages. A menu above the message area will display your options: Add, Delete, View, Sort, Print and Exit.

To make a selection, use the left and right cursor keys to move the highlight onto the option you want and press the return key. The highlight even wraps around from one side to the other.

ADDING REMINDERS

The first thing you'll want to do is add some reminders to the file. Select Add from the menu and enter the event date at the prompt. To help prevent typing in invalid information, the program accepts at this point only numeric input and real dates. For example, you can't list the date of someone's birthday as 11/31/86, since November has only 30 days; or if you try to schedule an appointment for February 29, 1987, the program will remind you that February has only 28 days in 1987. Try making it 1988 (a leap year), however, and it'll work fine.

After you type in the event date, you must specify the number of days in advance that you want to

be alerted to the event. Each time you use Reminder 128, the program retrieves only those events that are upcoming within the specified number of days.

Next, type in a message of up to 56 characters (with no commas or colons). Messages can be either one-time reminders that are automatically deleted once the date has passed, or annual reminders, for birthdays and such, that are automatically updated for the next year and written back onto the disk. To designate a reminder as annual, all you have to do is begin the message with an asterisk.

Once you've typed several reminders into your file, you can check to see if they're really there with the View option on the main menu. This displays all the reminders in your file along with their record numbers. It's a good idea to keep your reminder file on a backup disk.

OTHER OPTIONS

If you decide to eliminate a reminder, do it with the Delete option. Once you've typed in the record number to tell the program which reminder to erase, that reminder will appear on the command/menu line at the top of the screen. To proceed with the deletion, press the return key. However, if you have second thoughts, you can abort the operation by pressing the escape key. To use file space efficiently, record numbers that have been freed by the delete process are the first ones filled when you add more reminders.

The Sort option arranges all the reminders in REMFILE in ascending chronological order. Although the program can find the reminders for any given day regardless of their order in the file, I find it reassuring to see them printed out in chronological order.

If you want a hard copy of your reminders, select Print from the menu. A small pull-down menu will appear, offering the choice of "Today's," for a printout of today's reminders only, or "All" for a listing of your complete reminder file. To move the highlight from one selection to the other, use the up-and-down cursor key, and then press the return key to begin printing.

When you're done using Reminder 128, leave the program by selecting Exit from the menu and answering Y to the prompt, "Leave Program, Are You Sure?" If, at the last instant, you remember an upcoming appointment you forgot to add, enter an N to return to the menu.■

Patching Up RUN Script 128

By Robert Rockefeller

RUN It Right

C-128; disk drive

This easy-to-use patch program can make RUN Script 128's printing capabilities even more useful to you. The patch provides a page ordering (.po) command, which allows printing on both sides of a page, along with a multicolumn capability for generating documents such as newsletters.

GETTING STARTED

To use the patch, first format a blank disk, then copy each of the following RUN Script 128 files onto it: C-128 Character Set and OB.RS NMI, from the "RUN Script 128" article in the December 1986 issue of *RUN*, and the Patch RS128 program. I'll refer to this as disk 1.

Remove disk 1 from the drive and insert a disk, which I'll call disk 2, containing a copy of the Boot RUN Script 128 program from the December article. Load the boot program, list the last line (280) and change it to read as follows:

```
280 BANK 1: SYS 1024
```

After making the line change, remove disk 2 from the drive, place disk 1 in the drive again and save the boot program to disk 1. Without this line change, none of RUN Script 128's print commands will work.

Now load and run Patch RS128 from disk 1. When you're asked to place in the drive a disk containing a copy of OB.RS128 2.40 (55 blocks of machine language) from the December article, be sure that the copy has been modified with the Create ML program from December. When you're prompted to place a "save" disk in the drive, use disk 1.

After you're finished, disk 1 will contain the following files: C-128

Character Set, OB.RS NMI, Patch RS128, Boot RUN Script 128 and OB.RS128 2.40 with the patch installed. To activate the revised RUN Script 128, just load and run Boot RUN Script 128.

The .po command must be followed by three numbers, which are separated by commas. Using the command requires a little planning on your part, to determine what parameters to use for your desired result, and it works only when continuous output is selected for the printout. Here are some examples that show how the .po command works.

PRINTING ON BOTH SIDES OF A PAGE

One combination of .po settings, .po1,2,1 and .po2,2,1, lets you print on both sides of the page, with text extending across the full width of the page.

The first setting is for the first pass. The first parameter in this setting (1) sends page 1 to the printer or disk; the second parameter (2) specifies that every second page after that also will go to the printer or disk. For example, the odd-numbered pages might be printed and the even-numbered pages not—a phenomenon I call "page cycle." The last parameter in the first setting (1) indicates how many columns will be output to the page before the page number is increased. It should equal the number of columns across the page.

The second setting is for the second pass. Notice that the last two parameters (2 and 1) are unchanged. The new first parameter (2) sends the even-numbered pages to the printer. You print on the same paper as in the first pass, but now you use the back side. If you wish, you can define different headers or footers for the odd- and even-numbered pages to place page numbers on opposite sides of the page.

PRINTING ONLY ONE PAGE

Another combination, .po20, 1,1.st21, joins the .po and .st commands to print only one page of a document. In this example, pages 1–19 will not be printed, but page 20 will. After page 20, the .st command will bring up the Next Output? prompt, at which point you can abort the print operation.

PRINTING A TWO-COLUMN NEWSLETTER

Only four .po settings are required to print a double-column newsletter on both sides of each page. This is a dynamic command that's useful for other purposes, too.

On the first pass, you'll print the first column on the front of the page. On the second pass,

you'll print the first column on the back of the page. On the third and fourth passes, you'll print the two offset columns.

The first page requires a command of .po1,4,2, plus a two-column format command of .pw39.lm6.rm1, both placed at the beginning of the document. Calculate the two-column format command by assuming that 80 characters will be printed across the page, and that the left and right margins will be set at 6. Four columns will separate text columns, resulting in 64 (80 – 6 – 6 – 4) characters per line. So, a two-column format will require each column to be 32 (64/2) characters wide. You can set dot commands I don't mention here, such as .tm, .pl, .hd and .hs, to any value you wish.

After placing the above command at the beginning of the document, print the document, rip off the paper and insert the back side of the first page in the printer. Print the second pass with the .po setting .po3,4,2, leaving the .pw, .lm and .rm settings unchanged at .pw39. lm6.rm1.

After the second pass has printed, insert the front side of the first page into the printer. If you're using a header or footer, I suggest you delete it and increase the .tm and .bm settings to compensate for the lines used by the .hd, .ft, .hs and .fs commands. Print the third pass with the setting .po2,4,2.pw80.lm42 .rm6 placed at the beginning of the document.

Remove the paper, then insert it for the fourth pass, with the opposite side facing the printhead and the setting .po4,4,2 .pw80.lm42.rm6 at the beginning of the document. Note that .pw, .lm and .rm are unchanged.

That's all there is to it! Your professional-looking, two-sided, multicolumn printouts will make a big splash—and no one has to know how easy they were to produce.■

Mega-Magic:

Fast C-128 Hi-Res Screen Dumps

By Jeffrey K. Goode

RUN It Right

C-128; Star-compatible printer

128 Hi-Res Dump contains machine language code that dumps hi-res screens to Star-compatible printers and includes two size options and a reverse video option.

Copy this program to a new disk. When you run it, a machine language file is created and saved to the new disk. Then, that ML file is automatically loaded and executed. At the Ready prompt, set up your graphics screen and load or draw a picture to print.

Turn on your printer, poke the column or reverse data (see options below) and use the SYS command for the size printout you want. It takes 3–4 minutes to print an 8½- × 11-inch picture vertically on the page (use SYS 4864) and about a minute to print a picture vertically on a quarter page (use SYS 4867).

You can position smaller pictures with POKE 4883,nc, where nc is the number of columns from the left margin to the bottom of the printed picture. Use a value of 0 to print a picture on the left side, 20 to center the picture and 40 to print the picture on the right side of the page. This Poke only affects pictures smaller than a full page, and it remains in effect until you change it with another Poke.

Control the Reverse Video option with a Poke to location 4887. If this location contains a zero (the default), the printout results in pixels turned on as black and those turned off (background) as white. Use POKE 4887,255 to reverse the color of the printed dots. As with the Column Position option, any change remains until you poke a different value.■

Retir'eze

By C. Frank Schulenberg

RUN It Right

C-64; C-128; printer optional

Retired already? Enjoying the golden years? Or is retirement still in the dreamy future? Whether you're retired or not, if your future is an album of hazy, out-of-focus pictures of mountain streams or a life in the sun, Retir'eze can either bring those pictures into financial focus or awaken you to the realization that you'll be a derelict on Skid Row within ten years.

Retir'eze works on both the C-64 and C-128 computers. It can be a boon to those interested in planning their retirement or those already trying to cope with the financial vicissitudes of retired life. It's easy to use, as it prompts you for answers to straightforward questions, then evaluates, computes and prints out your financial picture.

This is not an investment planning program. No mention is made of specific stocks, bonds or mutual funds. The program lets you enter the return on investment (ROI) and makes no reference to timing decisions for market entry or profit-taking flights. You'll notice that market insiders' jargon (such as the Dow, AMEX, futures, commodities and industrials) is absent. Instead, Retir'eze will inform you of the effects of yearly inflation rates on your retirement budget and show you how differing ROI percentages will affect your investments during retirement.

Retir'eze lets you alter financial variables that are certain to change before and during retirement. Other financial changes, such as unexpected purchases or lottery winnings, can be added to Retir'eze in the year in which they occur. You can compare this new financial data with earlier Retir'eze-generated statistics to provide an updated statistical base. Of course, you can repeat the process any number of times.

The printout generated by Re-

tir'eze ends automatically when any of the following conditions are met:

1. The *yearly budget* exceeds the total available *investment/income/benefit* resources.

2. The *invested funds* total exceeds six figures.

3. The program has processed 40 years of data.

4. The user specifies an ending year less than 40 years beyond the *retirement* year.

The completed Retir'eze computer printout always includes a record of input data, including inflation rates, ROIs and cost-of-living adjustments (COLAs). The program also summarizes, qualifies and prints dollar amounts and effective years of any financial data you've typed in.

When entering information in response to Retir'eze's questions, don't use percent signs, dollar signs or punctuation marks, such as commas. Decimals are permitted. For example, you would use 10550.73 to represent $10,550.73. You can answer most questions with a single digit, a series of digits or a Y or N.

It's best not to skip any of the questions. Based on your answers, however, the program will bypass unnecessary questions. For instance, an N answer to the first section of a multipart question will take you to the next question.

To help you understand what it is Retir'eze can do for you, see the profile of a fictitious couple nearing retirement that accompanies this article. R

The Testkases Look Toward Retirement

George and Mary Testkase are preparing for retirement. George will reach his firm's minimum retirement age of 62 in 1988 and will also be eligible for early Social Security benefits that year. His wife, Mary, has worked on and off during their marriage and has met the minimum qualifications for Social Security benefits. Mary is working now and plans on retiring in 1993. They have decided to set their yearly retirement budget at $24,000 starting in 1988, the year George will retire.

The Testkases have accumulated $41,783.44 in assorted savings accounts, IRAs and a mutual fund. Also, George has a partially vested interest in his firm's profit-sharing plan.

Their local Social Security office has determined that George's monthly benefits will be approximately $570 upon retirement, assuming he continues to work at his present salary until 1988. Mary's benefits when she retires will include earned benefits based on her own work record, plus benefits she is entitled to as a spouse. She expects her

1988—GEORGE RETIRES:

Investments/savings = $54,000
Yearly pension, Social Security, etc. = $14,933
Mary's earnings = $14,449
Gross INCOME/BENEFITS = $29,382

1991—GEORGE'S PROFIT-SHARING PAID

INVESTMENTS + $33,400

1993—MARY RETIRES, GETS SOCIAL SECURITY

Mary's earnings – <$15,500>
Mary's S.S. benefits + $3,690
Net INCOME/BENEFITS – <$11,810>

1994—VACATION YEAR

INVESTMENT withdrawal – <$5000>

1995—TEN-YEAR ANNUITY MATURES

INCOME/BENEFITS increase = $6000

1996—FINAL MORTGAGE PAYMENT

Yearly BUDGET decrease – <$4356>

2005—TEN-YEAR ANNUITY EXPIRES

INCOME/BENEFITS decrease – <$6000>

Table 1. The Testkases' financial retirement timetable.

total monthly Social Security benefits to be about half of George's.

They are assuming the average yearly inflation rate won't exceed 5 or 6 percent and are also counting on COLA increases from part of their yearly income for a 2 percent annual gain in their *income/benefits*.

George's firm permits early retirees to delay taking their profit-sharing until they reach 65. George expects his share to be about $33,400 net, if present business conditions prevail. He is reluctant, however, to count on this as a firm asset. Considering tax advantages and other factors, George has decided to wait until he's 65 to claim his share.

The couple's assets also include an inheritance-financed ten-year annuity with guaranteed monthly payments of $500 starting in 1995. In addition, they are looking forward to paying off their home mortgage in 1996.

The Testkases have long postponed their dream vacation: a month-long tour of Europe. They've now scheduled it for the year following Mary's retirement and have decided to allocate $5000 from their invested assets for the trip.

See Table 1 for a look at the Testkases' major financial retirement timetable. ■

Twiddle

By Michael Broussard

RUN It Right

C-128

Twiddle is a two-player strategy game for the C-128. It's written in Basic 7.0, except for a short machine language subroutine that's poked into the cassette buffer when the program starts.

The object of Twiddle is to wipe out all your opponent's pieces before he or she wipes out yours. The rules are deceptively simple; once you start playing, you'll discover more and more subtle and complex strategies. To play the game, simply load TWIDDLE 128 and type RUN.

The program will first ask whether you want to play the standard game. To choose this option, just press the return key, and the screen will blank out for a few seconds while the program sets up the board. (It's possible to vary the rules from game to game to make it more challenging for advanced players or to introduce a measure of chance. Exactly how you do this is explained later.)

Soon a five-by-five square grid will appear on the screen, labeled with the letters A through E on the left (one letter per row) and the numbers 1 through 5 across the top (one number per column). Each square is identified by specifying its row and column; for example, the square in the upper-left corner is A1, and the square in the lower-right corner is E5. Player 1's pieces are the red squares in the top row of the board (squares A1–A5); player 2's pieces are the blue diamonds in the last row (squares E1–E5).

THE BASIC MOVES

When the game begins, player 1 (the red squares) has the first turn. A turn consists of three moves, which can be made by a single piece, or split up among multiple pieces as desired. To specify a move, just type the location of the starting square, followed by the destination square, and press the return key.

For example, to move from A2 to B2, you would type A2B2

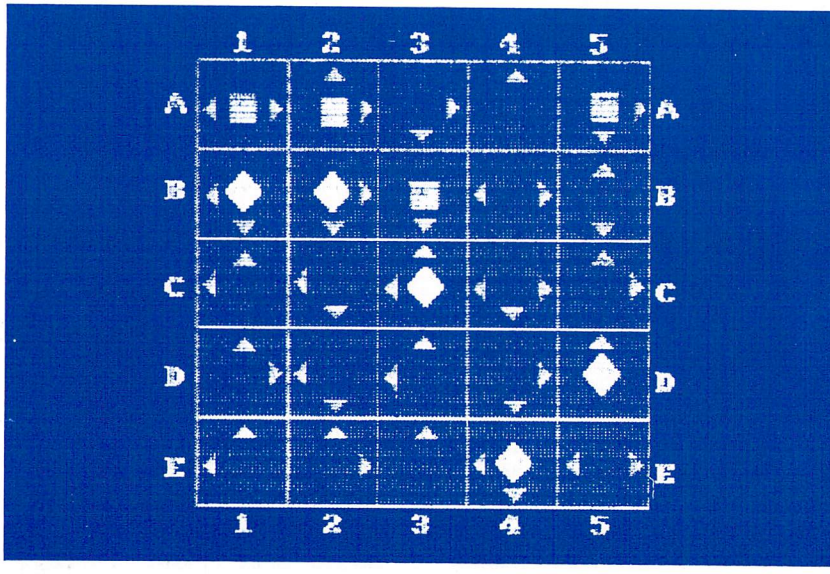


Figure 1. The standard Twiddle game board.

<return>. If you make a mistake, use the delete key to erase one character at a time. You must make the correction before you press the return key.

Figure 1, which shows a possible configuration of the game board, will help you to visualize the movement rules as they are described. Each square on the board contains from one to four arrows. The placement of the arrows is random and varies from game to game.

For a piece residing on a given square, these arrows indicate to which other squares that piece may move. For example, square B2 in the figure has two arrows, one pointing to the right and one pointing down. This means that a piece at B2 may move to either B3 (one square to the right) or C2 (one square down). It may not move to either of the other two adjacent squares, B1 or A2, because there is no arrow pointing in either of those directions. (Pieces may be moved only horizontally or vertically, not diagonally.)

If a piece is on the left edge of the board, say, at A1, it may "wrap around" to the right edge of the board (square A5, in this case) if there is an arrow pointing to the left, as in Figure 1. More

generally, if a piece makes a move that would send it off the grid, it wraps around to the opposite side of the board. As another example, the piece on A2 could move to A3, or it might wrap around to E2.

A piece may not move onto a square occupied by another piece of the same color. For instance, the piece at A1 may not move to A2. Moving onto a square occupied by an opponent's piece captures that piece and removes it from play (which is, of course, the object of the game). In Figure 1, the piece at B2 can capture the piece at B3. It cannot, however, capture the piece at A2, because there is no arrow pointing from B2 to A2.

RULES OF THE GAME

Much of the strategy of the standard game hinges on the following rule: When a piece moves out of a square, the arrows in that square rotate clockwise 90 degrees. For example, A1 contains two arrows, one pointing left and the other right. When a piece moves from A1 to A2 or wraps to A5, the arrows in A1 rotate one turn clockwise and now point up and down. The next piece that moves into A1 will have to exit by moving to B1 or wrapping to E1, as that's where the arrows now point.

If desired, you can "spend" a move by specifying the source and destination to be the same square. For example, "moving" a piece from C3 to C3 uses up a move but does not change the position of any other piece on the board. It does, however, make the arrows in C3 rotate.

There are two other rules in the standard game. First, player 1's pieces may not wrap around from the top row to the bottom row (even if the arrows in the square would allow such a move) until each of player 1's pieces has moved at least once. Likewise, player 2's pieces may not wrap around from the bottom to the top of the grid until they satisfy the same requirement. The purpose of this rule is to prevent one player from gaining an unfair early advantage by wrapping around on the first move and wiping out several of his or her opponent's pieces.

As I mentioned earlier, a turn consists of three moves. Starting with the sample diagram shown in Figure 1, suppose the diamond player elects to move as follows:

1. C3→B3 Captures the piece at B3 and makes the arrows in C3 rotate.

2. B1→B1 Makes the arrows in B1 rotate, but no piece changes position.

3. B1→A1 After move 2, an ar-

row points up from B1; this move captures the piece at A1 and makes the arrows in B1 rotate again.

Toward the end of the game, few pieces will be left on the board, and the players will be spending most of their time trying to stay away from each other. Twiddle contains a rule that is designed to ensure that the game *will* eventually end.

If 15 turns elapse without any piece being captured, the number of moves per turn increases by one, from three to four. If 15 more turns elapse, the number of moves increases to five, and so on up to a maximum of six. As the number of moves per turn increases, it becomes harder to stay out of reach of enemy pieces. The player who captured the last piece is the first player to have a turn with the increased number of moves.

ALTERING THE GAME PARAMETERS

That's all there is to playing the standard game. Since different players might wish to use different rules and difficulty levels, you can specify certain parameters at setup time. Remember that when the program starts, it asks whether you wish to play the standard game, and a default answer of "yes" is displayed. If you'd like to change one or more game parameters, press the cursor-right key to change to "no." Then press the return key, and you'll be asked to specify the size of the game board.

The default size, five-by-five, will be displayed, but again you can use the cursor-right key to loop through all the possible choices. (Use the cursor-left key to cycle backward.) You can choose any grid size, from three-by-three up to six-by-six. Press the return key to lock in your choice.

The next parameter you can set is the number of moves each player gets per turn. The default is 3, but you can change it to any value from 1 to 6.

You can also specify how the arrows should rotate when a piece moves out of a square. The default is one 90-degree rotation (clockwise), but you can specify random rotation, in which case the arrows turn 90 degrees anywhere from one to four times, or no rotation, which means they won't turn at all.

Finally, you can override the rule that forbids wrapping from the top to the bottom of the board (and vice versa) before all the pieces have moved at least once. Just cycle to "yes" when asked whether you want to allow immediate wraparound.

PERMANENT MODIFICATIONS

After playing the game several times, you might discover that you always begin by changing the size of the board or increasing the number of moves per turn. If you like, you can customize the program so that it uses your preferred start-up parameters as the defaults for the standard game. This is done by modifying the values of certain variables in line 10 of the listing. (Be sure to save the new version of the program after making any changes.)

For example, you might decide that you'd like the default game board size to be six-by-six instead of five-by-five. You can specify this by changing the value assigned to the variable BD from 5 to 6. From then on, when you choose the default setup, the board will be six squares on a side instead of five.

Here's a summary of the other variables you can change and what their values mean:

—RR specifies the amount of rotation applied to the arrows in a square when a piece moves out of the square. The default is 1, but you can change it to 0 (no rotation) or 4 (random rotation).

—The default number of moves per player is 3, but you can easily change this by modifying the value assigned to the variable NM.

—Another default is no wrapping from the top to the bottom of the grid until a player has moved each piece at least once. You can modify this by changing the value assigned to the variable WF from −1 to 0.

—During play, the number of moves available to each player per turn increases by one if 15 turns elapse with no piece on the board being captured. This is controlled by the value of the variable TX. Changing this value to 9, for example, would make the number of moves per turn increase if nine turns passed without a capture. If you never want the number of moves per turn to change, set this value to some extremely large number, say 99999.

—Finally, the default background color of the screen is green (6). You can alter this by changing the value assigned to the variable FC.■

Break the 128 Memory Barrier!

By M. Garamszeghy

RUN It Right

C-128; 1700 or 1750 RAM expander

The 1700 and 1750 RAM expansion modules for the Commodore 128 provide 128K and 512K, respectively, of additional random access memory. Basic 7.0 provides the Stash, Fetch and Swap commands to transfer data to and from the expansion memory, and the latest version of C-128 CP/M (CP/M 3.0) fully supports the expanders as RAM disks (drive m:).

However, Basic 7.0 lacks a RAM Disk mode. Stash, Fetch and Swap are not "true" RAM disk commands, because they don't reset the various start- and end-of-file pointers required to store and retrieve Basic and machine language programs.

This simple yet powerful RAM disk program gives the C-128 true RAM disk capability. It lets you store up to two 64K Basic program files in the 1700 RAM expansion module or up to eight 64K files if you have the 1750 module.

Keep in mind that the storage is temporary, because any data stored in a RAM expander is lost when you turn off the power to the computer or press the reset button.

The RAM Disk 128 program installs itself as a device driver in an unused area of memory starting at address $1300 in bank 0. The cassette buffer at $0B00 becomes a RAM directory buffer. The RAM disk is assigned device number 1, and it can't be used in conjunction with a Datassette.

Once activated, RAM Disk 128 will usually remain in operation until you reset the system by turning off the power, pressing the reset button or entering a SYS 57344 or SYS 57416.

RAM Disk 128 intercepts calls to the Kernal Load and Save routines

by changing the indirect jump vectors at $0330 and $0332. Control is routed back to the Kernal if the device number is 1; otherwise, RAM Disk 128 is directed to special Load/Save routines that access the RAM expander and reset the file pointers.

Each RAM block contains a directory entry, which consists of a filename, default load address and file length, in the first page of the block.

The direct memory access (DMA) process used by the RAM expansion controller (REC) chip works exclusively in Slow (1 MHz) mode. If RAM Disk 128 is called in Fast mode (2 MHz), it automatically switches the computer to Slow mode.

PUTTING THE RAM DISK TO WORK

Operating RAM Disk 128 is easy. RAM DISK LOADER is a short Basic program that creates a machine language file called RAM Disk 128. Save it to a new disk before running it. Once you've created the RAM Disk 128 machine language file, you can begin using it immediately.

To activate RAM Disk 128 from a cold start, first place the disk that contains the RAM Disk 128 file in your disk drive. Then, if your drive is a 1571, type BOOT "RAM DISK 128" <return>. If your drive is a 1541 or 1541-compatible, type BLOAD "RAM DISK 128" <return>, then SYS 4864 <return>. Either way, a message will appear telling you that RAM Disk 128 is active and what the RAM size is—128K or 512K. If no expander has been installed, an error message will appear. The RAM disk will instantly "format" itself, and a Ready prompt will appear.

RAM Disk 128 is invisible when activated, and doesn't impair the normal disk input/output functions.

SAVING FILES

Since RAM Disk is assigned device #1, files are saved to it using a syntax that resembles saving to a Datassette. Just type SAVE"n:filename" <return> to save a file. In this statement, n is a number from 0 to 1 (for the 1700 expander) or 0 to 7 (for the 1750 expander) that corresponds to the bank in the expander where the file is to be stored. Note that only one file per bank is permitted. The filename is optional, since the bank number is adequate to identify the file. If n: is missing, a Syntax error will occur.

A Save operation will replace the existing contents of the bank being saved to, whether it's a RAM Disk 128 file or data written to the expander via Stash or Swap.

DELETING FILES

Once a file has been placed in a RAM disk bank, it will remain until another file is saved to the bank. If you want to delete a file from the bank and leave the bank empty, clear the computer's memory with the New command. Then enter SAVE"n:" <return>, where n is the number of the bank to be deleted.

LOADING FILES

To load a file from the RAM disk, type LOAD"n" <return>, where n is the expansion bank to load. A Load Error message will appear, but don't worry about it—the file *will* load. Attempting to load from an empty bank will produce a File Not Found error.

You can display the directory entry for each bank by typing LOAD <return> or LOAD"$" <return>. If a bank number is not listed in the directory, that bank is empty. The directory will not overwrite a program in main memory, unlike the LOAD"$",8 command.

When you're using the RAM disk, you can't use BLoad and BSave with machine language programs. You must load and save these programs with a C-128 monitor designated as device #1.

Verify, Open, Print# and other file-handling functions are not implemented on the RAM disk, but they're still available for use with disk and serial bus input and output.■

Mind Your Mortgage

By Robert Kupfer

RUN It Right

C-128

If you're planning to purchase a home or refinance the one you have, you need all the information you can get to make sound mortgage decisions. Such information can also be helpful with the mortgage you have, by revealing how much the loan is really costing you and how you might be able to save on it.

Loan Analysis 128 will tell you everything you need to know concerning the payments, interest rate and repayment period on a mortgage. It will also provide a month-by-month amortization schedule for the life of the loan, which, in addition to the items mentioned above, will include your cumulative interest, cumulative equity and a breakdown of your monthly payments into interest and principal portions. At the end of every year, the program will calculate the amount of interest you paid, which is important for tax purposes, and the amount of equity you built up for that year.

Loan Analysis 128 will work on any fully amortized loan for which interest is calculated on the remaining balance. This covers just about all loans that are currently offered by financial institutions.

A SAMPLE LOAN

Make sure you've set your monitor to 80 columns before running the program. Now, let's say you're buying a home for $100,000 and are considering an 11.5 percent mortgage over 30 years. Enter these figures when the program asks for them, being sure not to include any commas. Then a display will appear that includes the loan amount, interest rate, monthly payment and total cost of the loan over its life span. These calculations are done in Fast mode.

Then the program will ask if you want to see an amortization table. For this example, press Y. The screen will clear and an amortization table for the first 12 months of the loan will be calculated and displayed. The sum of the two columns marked INTEREST PAYMT and EQUITY PAYMT equals your monthly payment. Note that, for quite a few years, the interest portion of the payment greatly exceeds the equity portion. To get the next year's amortization schedule, press the return key; press any other key to quit.

Loan Analysis 128 is designed to be used with an 80-column RGB monitor. However, you can use the program with 40 columns if you're willing to do without some of the amortization table displays.

A MONEY SAVER

You might want to use the amortization table to determine how to repay your loan sooner. Perhaps you could save thousands of dollars in interest, while paying little additional money each month. For example, say you're about to make payment number 5 on your loan. Make out a check for that payment as usual, then check the amortization table to see what the equity portion of payment number 6 is. Make out a separate check in that amount, and print on the back of the check, "To be used toward principal only." Enclose this check with the regular payment for month 5.

When you're ready to make payment number 6, send in the full amount as usual, plus a separate check for the equity portion of month 7, and so on, as long as there is any principal balance outstanding. What you're doing is reducing the amount of the principal that the interest is calculated on, thus reducing the total cost of the loan and the length of time it'll take for you to pay it off in full.■

The Light Choice

By Bob Guerra

RUN It Right

C-128

These days, practically all commercial computer programs are menu-driven, but different programs provide different ways to select the option you want. Some have you enter the first letter of the option, such as P for print, S for search or Q for quit. Others number the options and ask you to enter the number of your choice.

For most Basic programmers, these two methods have become standbys. A slightly more sophisticated approach uses a joystick- or mouse-controlled arrow to point to the option, but this approach has to be written in machine language to make the arrow travel smoothly and quickly around the screen.

A third method, possible to code in Basic with only a little more effort than the letter and number systems, is the moving highlight. Here, the options are displayed across a command line at the top of the screen, and you use the left/right cursor key to make the highlight move from option to option. When the highlight covers the option you want, you press the return key to send execution to that routine.

Although the moving-highlight method requires as many or more keystrokes for making a selection as the first-letter method, it turns out to be faster (especially for nontypists), because you always press the same key to make a choice. Another advantage of this approach is that it gives programs a polished, professional appearance.

Also, when you program a moving-highlight menu for the C-128 in 80-column mode, there's plenty of room across the top of the screen for menu items, and the Window command makes it easy to create "pull-down" submenus from the main selections.

GET ROUTINES

Here's how the highlight system works. As the menu is initially displayed, the first option (on the

left) appears in reverse lettering, and the rest are in normal lettering. This makes the first option look highlighted. A Get routine waits for you to press either the left (CHR$(29)) or right (CHR$(157)) cursor key, so it can decide whether to move the highlight to the next option on the right or jump it to the last option on the right.

What actually happens when the highlight "appears" to move is that the two options involved are rewritten—the one that was originally in reverse lettering is rewritten in normal lettering, and vice versa. Because the rewriting happens so quickly, it looks like the highlight changes position. Once you've moved the highlight, a second Get routine made for the new situation is executed.

To make the whole system work, a separate four-line Get routine is needed for each of the menu items. These routines reside in lines 50–320 of the MENU HIGHLIGHTER. The first line of each routine gets the string variable X$, checks to see if its value is CHR$(29) (cursor-right) and, if it is, rewrites the two options and sends execution to the appropriate routine. If the value of X$ isn't CHR$(29), execution falls through to the second line to see if X$ equals CHR$(157) (cursor-left). Again, if it does, the options are rewritten and execution goes to the appropriate routine.

If X$ isn't CHR$(157), execution proceeds to the third line, where the variable is checked for CHR$(13), the return key. If it is, execution branches to that part of the program named in the menu option that was highlighted when you pressed return.

If you didn't press an appropriate key, execution falls through to the fourth line, which loops back to the first line to get X$ all over again. Thus, the program ignores all keystrokes other than cursor-right, cursor-left and return. An advantage to using multiple Get routines is that they enable you to hold down the cursor keys and rapidly cycle through the various menu choices in either direction.

PULL-DOWN MENUS

You can create pull-down submenus by defining a small screen window in the area directly below a main-menu choice, then listing the options in a column within the window. This time the highlighted option is the only one *not* written in reverse, and the program checks the keystrokes for CHR$(145) or CHR$(17) (cursor-up or cursor-down).

To see an example of a pull-down menu, select option 4 in the main menu of the program.

The code for this submenu resides in lines 4000–4200 of the listing. When you're programming pull-down menus, remember to clear and redefine the window immediately after a selection has been made.

Although highlight-type menus are usually associated with productivity applications and utilities, they work equally well with games, and if the game you write is largely joystick-controlled, you can easily use it to move the highlight and make selections. Instead of getting a keystroke and checking to see if it was a cursor key, you check the status of the reserved variable—JOY(1) or JOY(2), depending on which port the joystick is using.

A value of 3 means the stick has been moved to the right, and a value of 7 indicates a move to the left. A value of 128 or higher means the fire-button—which has the same function as the return key—has been pressed. If you wanted to allow both keyboard and joystick selection, you certainly could combine both methods. ■

Add/Calc-128

By George Noeth

RUN It Right

C-128 (in 80-column mode); printer optional

This little calculator program, written in Basic 7.0, is designed to turn your fancy new C-128 system into a rather simple printing calculator. I wrote Add/Calc-128 to learn about Basic 7.0, so I used as many commands as possible, as well as the 80-column screen and the numeric keypad. As a programming challenge, I also avoided the Peeks and Pokes I normally use.

When you run Add/Calc-128, it first asks if you intend to use the printer; respond Y or N. Then a screen appears, displaying two windows and the program instructions.

Press a number on the numeric keypad. You'll hear a tone and the number will appear in the left window. If it's a negative number, it'll be in red. To clear the window, press the F5 key. If you make a mistake entering a number, hit the F5 key and then reenter the number.

Pressing the + key moves the number to the right window, clearing the left window in the process. As the right window fills with figures, they scroll vertically out of the window, with the most recent entries remaining in view.

To total a column of figures, press the return key after the last entry. To multiply two numbers, press the F1 key after the first entry, then the return key after the second. For instance, the proper entry sequence for multiplying 25 by 50 would be 25, F1, 50, return. The sequence for division is the same, except you press F3 instead of F1.

The program retains the result of an operation to use in the next. For example, if you press the + key after taking a sum, that sum will appear on the screen as the first entry for a subsequent multiplication.

If you're interested in exploring Basic 7.0, you might begin by

adding some enhancements to Add/Calc-128. There's plenty of room for improvement, because it lacks most of the features of the more sophisticated printing calculators. It could use multiple memory keys and a percent key, for instance. Whether you're an old pro or a novice like me, I'm sure you'll find the C-128 and Basic 7.0 an extremely versatile and easy-to-use combination. ■

64 Notepad Update

By Bob Kodadek

RUN It Right

C-64; printer optional

In my article, "Programmers, Take Note!" (*RUN*, September 1986), I introduced the 64 Notepad program, a desktop accessory that provides an instant-access text window for typing in and recalling programming (or other) notes without affecting the screen display. Notepad is RAM-resident, interrupt-based and transparent to most programs. When you access 64 Notepad, it "freezes" the program so you can enter notes or other information.

Now I've added two routines to the program. NOTEPAD 1 saves and loads any Notepad window. Special file identifiers are appended to all filenames, a fresh workscreen is provided, and the error channel is read and displayed after each disk operation.

An accurate digital time display is also included, using one of the 64's time-of-day clocks. NOTEPAD 2 prints out Notepad windows (while ignoring text outside windows) and full screens. These commands are available at the touch of a key whenever the Notepad is open, even while another program is running.

I've supplied the patches in the form of Basic loader programs, which you can append to the base 64 Notepad with the accompanying program, NOTEPAD 3.

ENTERING THE NEW ROUTINES

To use the new Notepad patches, you need a working copy of 64 Notepad. NOTEPAD 1 contains the machine language for the Save, Load and Digital Clock routines. NOTEPAD 2 contains the printer routines. Keep in mind that these are temporary files used to create one large Notepad program. They do not perform any function until they're appended to the main 64 Notepad program.

You can use NOTEPAD 3 to append one or more Basic programs from a disk to a program in Basic memory. Just be sure

that the programs you want to append to the resident program have higher line numbers. Remember to save the program before executing it, since it erases itself when run.

The proper syntax for an append is: SYS (SA),"FILENAME". The variable SA is the starting address of the Append routine. The machine language code is completely relocatable; to place it into a different memory location, change the value of variable ML in line 10 of NOTEPAD 3. You'll be using it at the present default location of 828, which is located in the cassette buffer.

Now you're ready to append NOTEPADS 1 and 2 to 64 Notepad. Load and run the Append program. During the next sequence of commands, always check to see that you have the correct disk, with the necessary program, in drive A (device number 8), before pressing return. In Direct mode, type in the following commands:

```
LOAD"64 NOTEPAD",8
SYS 828,"NOTEPAD 1"
SYS 828,"NOTEPAD 2"
SAVE"64 NOTEPAD II",8
```

If all goes well, the OK message will be displayed after each append, and you'll now have a finished copy of the expanded Notepad program.

USING THE NEW ROUTINES

Shortly after running the new program, you'll be asked to enter the correct time. The first input prompt will request hours, and a second prompt, minutes. Since this is a time-of-day clock, the hours must be any number from 1 through 12. By pressing only the return key in response to these two prompts, the internal clock will start at zero and may be used as a timer. The current time is always displayed in the lower-right corner of the text window. Although the digital time display stops during printing, the clock will still keep the correct time. Consequently, the display is updated automatically when the printing operation is complete.

After the Ready prompt appears, you can use the Notepad. Press the CTRL-O combination to open the window. The new routines will take over the function keys whenever the window is open, and they will perform the following functions:

F1: Saves a Notepad window.
F3: Prints the window currently in memory.
F5: Prints a full screen.
F7: Loads a Notepad window.

Press CTRL-C to close the window, and the keys should revert to their original functions.

When you wish to save (F1) or load (F7) a window, one of the

prompts, SAVE: N. or LOAD: N., will appear. Now give the program a filename. Your notes will be safely stored, and you'll have a fresh workscreen to write in. To abort at any time, either press the stop key or press return without entering a filename. The N. in the prompts is a Notepad file identifier, and it automatically becomes part of the filename. There's no need to type in this prefix; it is done for you. Remember that you are limited to 14 characters in the filename because of the prefix. The Custom Input routine, complete with cursor and delete functions, also limits your input to 14 characters.

Each time you save or load a window, the error channel is read and displayed on the screen. Press return, and your text will appear. These added commands also make it possible to use the 64 Notepad as a simple index card file. A single-sided disk can hold up to 144 Notepad files and is only limited by the directory.■

RUN Script 128, Part 2: Defining Printer Macros

By Robert Rockefeller

RUN It Right

C-64 or C-128 in C-64 mode; disk drive; printer

Define Macros, the C-64-mode program that accompanies this article, cannot itself be run in C-128 mode. However, the printer macro table that the program creates can be used with RUN Script 128 2.40.—**Eds.**

Many Commodore owners use non-Commodore printers such as the Okimate 10 or Epson MX-80. These printers often have desirable features, like italic character sets and the ability to do underlining. The printer-macro feature of RUN Script enables you to customize your copy of this word processor so you can take full advantage of whatever capabilities your printer may possess.

Printer macro character strings are always sent to the printer when output is to the screen. This is necessary when output is being switched between the screen and printer. An idiosyncrasy of RUN Script 128 2.40 is that the printer must be turned on when output to the screen is taking place, because a file is always opened to the printer when you select output to the screen. If RUN Script ever seems to "hang up" mysteriously during a printout, check your printer.

DEFINING PRINTER MACROS

You may select any upper- or lowercase alphabetic character to be a macro character. You then create a table of printer macros with the Define Macros

program. Each macro character represents a string of user-defined characters. When a macro character is encountered during printing, this string, rather than the macro character itself, will be sent to the output device.

For example, let's say you own a printer that requires the sequence ESC X (decimal values 27 and 88) to start printing double-width characters. With Define Macros, you can select a character—D, for instance—to represent this two-character string. Then, when D is encountered during printing, the decimal sequence 27,88 will be sent to the printer to produce double-width characters. You could define another character, perhaps d, to represent the sequence to stop printing double-width characters.

This macro feature is most useful for printing titles and subheadings. To create a double-width heading, first place the cursor in front of the heading, then press the F3 key. A "*mac*" message will appear on the status line. Next, press the upper- or lowercase alphabetic character you've chosen to activate the double-width capability (in my example, D). Finally, move the cursor to the end of the heading, press F3 again, and press the key you've chosen to deactivate the double-width feature (d, in my example). That's all there is to it! If you press any nonalphabetic character, the operation will abort.

Your table can consist of 52 different macro definitions, each of which can be from one to 20 characters long. I've allocated exactly 500 bytes in memory for the complete macro table.

CUSTOMIZING MACROS

Before running Define Macros, make a list of the alphabetic characters you want to represent the various functions your printer can handle. These will be your macro characters. Beside each macro character, write the decimal values of the character sequence that must be sent to your printer to implement the function each individual macro character represents. Then run the Define Macros program.

First you'll be prompted to select a macro character. Enter any upper- or lowercase character from A to Z. (If you make a mistake and wish to cancel a macro definition, use the * key.) You'll then be asked how many characters will be represented by the macro character you've entered. Count them from your list and enter the total.

Next, enter the decimal value of each character in the string, starting with the first and continuing until all have been entered. Once you've done this, you'll

have defined one macro. The prompt, "finished all definitions (y/n) ?" will then appear. If you have more macro definitions to enter, type n and press the return key.

After you've entered your list of macro definitions, press the y key at the prompt. Within seconds, the program will create the table of macro definitions, then prompt you to save the table to disk and provide the proper device number.

When you're in RUN Script 128 2.40, you can easily load in your macro set by pressing F1, followed by m.■

Word Wars

By John M. Smyczynski

RUN It Right

C-64

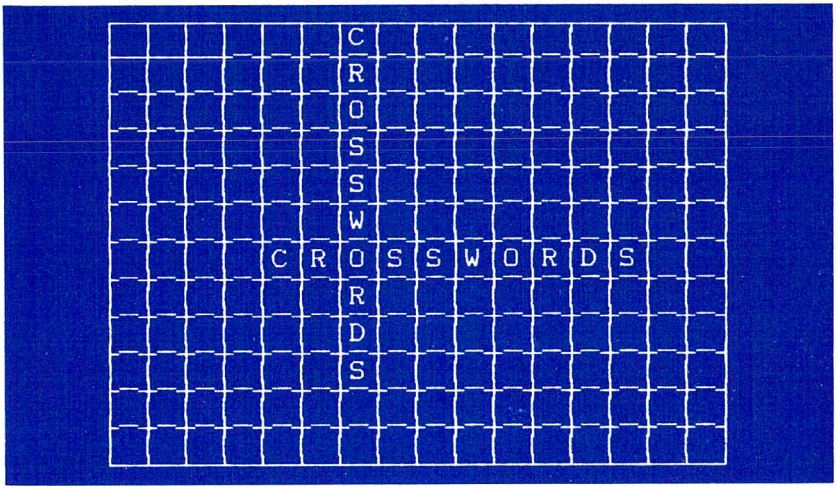
Crosswords is a challenging game in which the players build words from a random assortment of letters and assemble them into a crossword puzzle. Each player accumulates a score based on the location of the letters he or she places. Designed to be a family or party game, Crosswords can be played by up to eight people, but also has a single-player option.

The game starts with a flashing introduction accompanied by random music, then a short pause while the program reads letters into an array and mixes them. Next the program asks for the number of players, their names and, if there's more than one player, a two-key code for each. Finally, it asks how many rounds the game will have. The preliminaries over, just press any key to start play.

BUILDING WORDS

As play opens, Crosswords randomly selects and assigns to each player eight letters from the mixed-letter array. Then a display appears, consisting of a blank crossword grid with a large, white cursor in the center. The upper-right section of the screen displays the active player's name, letter assortment and current score; the middle-right section reveals the current high score, the number of the round and the game prompts; and the lower-right section shows the name of the next player (if there is more than one player) and his or her letter assortment.

The first set of game prompts (in a white background) are for moving the large, white cursor. Decide what word you want to build out of your eight letters, employing as many of them as you wish. Then use the cursor keys to place the cursor where you want your word to start and press H (horizontal) or V (vertical)



to specify the direction in which the word will extend. Only in the first turn of the first round of play can the word start anywhere on the puzzle grid. Subsequent words must link to one already there. This is done by crossing them or by placing them adjacent to each other. A word can link to more than one existing word as long as all those involved are valid.

After you've selected H or V to specify the direction you want your word to run, a set of spelling prompts (on a blue background) will appear. Press the number keys 1–8 to designate, in order, the letters you want to place on the grid. As you place a letter, it will disappear from your assortment.

If you make a mistake, press D (delete) and a third set of prompts (in light green) will appear. The W (word) option in this set of prompts erases all the letters you've just placed on the puzzle and returns you to the cursor-movement prompts. W is useful not only if you make a mistake, but also if you spot a better place to start your word. L (letter) erases the last letter you placed and returns the spelling prompts. The letters you erase with W or L reappear in your letter selection. If you decide you don't want to erase any letters after all, press N (none) to go directly back to the spelling prompts.

When you've finished building your word, press return. If you haven't linked your word to another one, a buzzer will sound at this point. If you have, the program will tell the next player

to enter his or her two-key code and okay your word. (A single player must enter only the okay.)

A word is not valid if it's not a real word, if it's misspelled or if it renders invalid the word or words it's linked with. If the next player thinks your word is invalid and presses N (no), your spelling prompts return so you can fix it. If the next player presses Y (yes), the scoring routine takes over.

As your letters are scored, their colors change depending on their value, a bell tone sounds and your cumulative score is updated. When the scoring routine is done, the puzzle reverts to its original color scheme, you get enough new letters to replace the ones you used, and the next player's turn begins.

If your assortment of letters or a crowded puzzle makes it impossible to spell a word, press P (pass) to skip your turn and get a whole new set of eight letters for your next turn.

SCORING

At the end of the game, the players will have filled in a crossword puzzle together, but the real object of the game is for each player to score as many points as possible. Generally, you score one point for each letter you place on the puzzle and one point for any letter you link to. However, you get a bonus when you place a letter or letters so as to make new words from existing ones.

I'll use a sample word sequence to illustrate the scoring. First, a player spells BOTTOM horizontally, for a score of 6. As scoring takes place, the letters change from black to white. The next player adds S, scoring 7, and the first six letters change back to black and the last letter changes from black to white.

The following player spells NOON horizontally, starting under the first O in BOTTOM. This player scores 12—2 apiece for the vertical ON, TO, TO and ON and 4 for NOON itself. As scoring takes place, the letters involved continue to change color, with red, cyan and so on indicating when letters are playing multiple roles.

If player 2 had spelled HERS vertically with the S at the end of BOTTOM, his or her score would have been 11—7 for BOTTOMS and 4 for HERS. The S would have become red, HER would have become white and BOTTOM would have become black. The S in this example does double duty.

When you add two or more letters to a word, they score double even if they can't stand alone as a word. The same is true if you place one letter at

the beginning and one at the end of a word.

A high score doesn't depend as much on your letter selection or how many you use as on where you put them in relation to other words. In planning your strategy, try not only to maximize your score, but to stymie your opponents as well.

FLEXIBILITY

Players need to agree on certain rules before starting a multiplayer game. For instance, they must decide whether contractions, foreign words, proper names, and so forth, can be used. Another issue might be how, if at all, a player should be penalized for placing an invalid word or for incorrectly challenging a valid word. The group might also want to set a time limit for turns, or make up teams.■

Mega-Magic:

Short Sorter

By Wilfred H. Protig

RUN It Right

Any Commodore computer

Magic trick $30F (*RUN*, August 1986) was a Basic number sorter. It was a handy program, but I've re-written it into a powerful programmer's tool.

The original program simply uses an integer array to count each value. My revision scans the array to be sorted to find the maximum value. That value is found and stored in AX; then an extra array is dimensioned for the sort. A second pass through the original array increments the F% array element to equal the maximum value. This is the reason the array must be dimensioned to the maximum value in the original array. The sort is now finished, but before the job is complete, the data must be placed in an array that Basic can use.

Simple lists can use the original array as shown in SORTER 1. Once the data is stored back in the original array *in order*, the sort is complete.

SORTER 1 reveals several limits imposed by this sorting method. First, the values must be positive integers. Second, it's impractical to use this method to sort values larger than 1000, because a large amount of memory will be used in the sort array. For example, to sort an array with a maximum value of 10,000, you have to use DIM F%(10000). This requires 20,009 bytes of memory and adds considerably to the time needed to create the array of sorted values.

Since two-dimensional sorts are more valuable than simple lists, SORTERS 2 and 3 are designed to print curves on printers that move paper in one direction. SORTER 2 sorts curves in which both Y% and X increase—curves that travel right and down, never left or up. SORTER 3 sorts by Y%, with X values going up or down, creating curves similar to a sine wave, for example.

Three extra arrays are used in SORTER 3. The first is the F% array, used for the sort. The other two are for the sorted values of the

Y and X coordinates. The Y% array is sorted and stored in the G% array; the X array in the H array. To speed the filling of the G% and H arrays, the last value in each original array is moved into the location of the last sorted element. This shortens the loop each time the subroutine at 160 is called, but the Y% and X arrays are scrambled during the sort. About 50 seconds is required to sort 100 pairs of data (range of 1 to 99 for Y).

The "read" in line 40 is used to read the Data statements. Simply change it to an input if you want to enter data from the keyboard. ■

Keeping Up to Date

By Michael Martone

RUN It Right

C-64; printer

Whether you need to organize your schedule or just want a calendar to tack onto the wall, Calendar Generator will provide one in a matter of minutes. In fact, it will print out a calendar for any month in any year (in the Gregorian calendar system) that fact or fancy might dictate. You could use it to plan a vacation in 1988, as a study aid in a history course or to find out on which day of the week New Year's Day 2000 will occur.

Calendar Generator will work with any printer that emulates a Commodore 1525 or 1526. It sends only a few control characters to the printer, and all of them are standard for most printers. Note that the CHR$(14) in line 300 turns on the enhanced (double-width) characters, and the statement PRINT CHR$(12) in line 385 is the Top of the Page command.

MAKING A CALENDAR

After loading the program, type RUN.The title screen will appear. Now load your printer with paper and turn on the power. Set the printhead at the top of the page and hit any key to get the Input prompt.

Now enter the month and year you want to print out. The month must be in two-digit format, and the year must be in four-digit format. If you want a calendar for June 1987, for example, enter 06,1987. *Don't* enter 06,87, or you'll get a calendar for June in the year 87—1900 years ago!

The printer will now turn out a calendar for the month you've specified. When it's finished, the program will ask if you want to print another calendar. If you respond Y, a new prompt for a month and year will appear; otherwise, execution will end.

Happy New Year!■

Solving the Split-Word Problem

By Ray Wright

RUN It Right

C-64; C-128 (in C-64 mode)

Most word processors have a feature called word wrap. This prevents words that appear at the end of a line from being split and continued on to the next line. Instead, the program carries over the entire word and begins the next line with it. This makes the text much easier to read, especially for someone who is unfamiliar with a computer's tendency to split words at the end of a line.

Unfortunately, this word-wrap feature is difficult to incorporate into a Basic program. Using a simple Print statement, you can make sure no words are split by properly placing extra spaces within the statement to be printed. But this will work only if the statement contains no string variables that may be assigned values with different lengths.

Consider, for example, the following simple program:

```
100 PRINT"{SHFT CLR}"
110 INPUT"NAME";N$
120 PRINT"{SHFT CLR}HELLO
    THERE, " N$," HOW ARE YOU
    TODAY? FINE, I HOPE! NOW
    LOOK ";
130 PRINT"CAREFULLY, "N$", AT
    THIS TEXT, AND SEE WHETHER
    ANY OF THE WORDS ";
140 PRINT"HAVE BEEN SPLIT
    ACROSS A MARGIN.":GOTO110
```

No matter what name is input in this program, there will be at least one line with a word cut off at the end.

This article's accompanying machine language subroutine, which you can easily include and use in any Basic program, solves the split-word problem. It uses memory in the range 51968 to 53236 ($CB00 to $CFF4). SPLIT WORD1 should be placed at the beginning of your Basic programs. Load and run SPLIT

WORD1. This will then load SPLIT WORD2 into memory.

Type in RUN 100 and press the return key. When the program asks for a name, type in names of various lengths, pressing the return key after each one. Notice that words are never cut off at the end of a line, no matter how long they are.

If you look carefully, you may be able to see the subroutine reprinting the text; it happens in a fraction of a second. In case you're curious, here's how it works.

First, you'll find that your Basic program prints the left arrow at the beginning of the text as a marker to show the subroutine where to begin. Then, when the subroutine takes over, it begins by printing an invisible marker (a reversed left arrow, which is the same color as the background screen color) to show the subroutine where the text ends.

It then searches through the screen memory (bytes 1024 to 2023) for the first left arrow. When it finds the arrow, it transfers to a block of memory beginning at 51968 ($CB00) the CHR$ value of each character of text to be reprinted (the block of memory is below the location where the subroutine itself resides).

Next, it reprints back onto the screen, one line at a time, the text from the memory block, deciding for each line which word should be the last one. When the entire text has been reprinted in this way, the subroutine returns control to your Basic program.

Now to use the subroutine in your program.

1. At the beginning of the statement to be printed, right after the first quotation mark, but before the first word to be printed, place a left arrow (←). (This arrow will not be visible on the screen; the subroutine will print over it.)

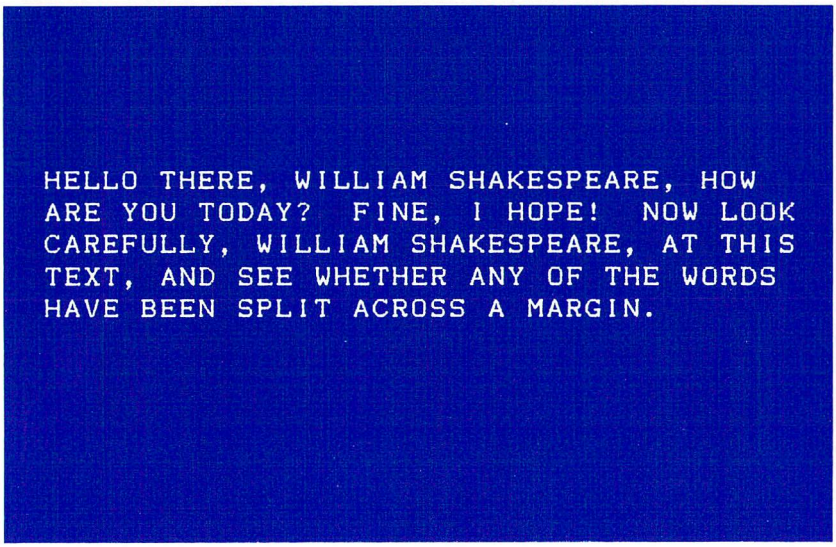
2. At the end of the statement to be printed, after the closing quotation mark and colon, type SYSL. (L represents the beginning address of the machine language subroutine; set it equal to 52992 in the beginning of your program.)

That's all there is to it! The computer instantly takes care of everything else involved in properly formatting the line.

LIMITATIONS

As you incorporate this subroutine into your programs, you must keep in mind a few of its limitations.

1. Since the left arrow and reversed left arrow are used as markers in this subroutine, you should not include these characters in your text (or allow left arrows to appear anywhere on



the screen), except for the one used as a beginning marker. Any other character may be included in the text.

2. The subroutine does not reprint text in multicolor. It will reprint everything using whatever color was last being printed before the subroutine was called.

3. The subroutine will not print reversed characters.

4. The subroutine will not print blank lines. If your Basic program prints two lines separated by a blank line, and the subroutine is applied to both lines at once, it will move the second line up so that it's right below the first. Therefore, if you need a blank line between two parts of your text, you must use the subroutine twice, separately for each part.

5. Make sure you do not include a SYSL in your program if you did not insert a left arrow in the text; if you do, the results are messy and could cause a program crash. Also, make sure that L (or whatever variable you choose to use after the SYS) is kept equal to 52992 throughout the program.

6. There may be rare cases in which, as the subroutine reprints the text, it does not completely print over the end of the old text that was on the screen before the subroutine was called. This

is unusual, since the word-wrap process almost always makes the text *longer* than it was originally. It makes it shorter only if you use the subroutine on a part of the text that includes a blank line or many adjacent spaces.

There is a way, however, to prevent the "old text" from being visible, even if part of it does remain on the screen. Simply have your Basic program print the text using the screen color (include the symbol for this color between the beginning quotation mark and the left arrow), and then, before calling the subroutine, have it begin printing the color in which you want the text to appear (put the symbol for this color right before the *ending* quotation mark). The demonstration program (SPLIT WORD2) includes lines 200–245 to show how this is done; to see how, enter RUN 200.

Notice that when you use the same color for both text and screen, the transition from being printed by Basic to being printed by this program is very smooth.■

Hook Up to a Portable

By Howard Parnes

RUN It Right

C-64; Model 100; 1650 or 1660 modem

I am sure that many dedicated Commodore users have been itching for Commodore to introduce a small, battery-operated portable computer that would be compatible with the C-64. Commodore's model SX-64 is an excellent "trans-portable," but it requires an electrical outlet to operate. Now C-64 users who also own a Radio Shack Model 100 can experience true Commodore portability.

I've written a machine language program that "creates" a portable, battery-operated Commodore by linking a C-64 and Radio Shack Model 100. Called the Model 100 & 64 program, it allows you to create text files on the Model 100, then transfer the files to the C-64 and save them on disk. Later, you can reload the files using one of the popular word processing programs for the C-64.

The two devices can communicate over telephone lines, or they can be linked directly. The only accessory necessary is a modem for the C-64; the Model 100 has its own built-in, 300-baud modem and software.

The Model 100's text editor operates like most word processing programs for the C-64. It offers most of the conventional block moves, along with search, text editing and cursor movement features. Using the 100 & 64 program, you can easily emulate the standard Commodore word processors on the Model 100, then upload your text files to the C-64.

SETTING UP

Load M100 TO C64. After you type RUN, you must specify the type of modem you're using with the C-64—a 1650 or a 1660/300. The program will set the appro-

priate parameters. If you're using a 1650-type modem, remember to set the originate/answer switch to originate, the half/full duplex switch to full and the telephone/line switch to line. With a 1660/300-type modem, you need only set the originate/answer switch to originate.

On the Model 100, you must set the originate/answer switch to answer and the communication status to M8N1D. The M stands for modem, the 8 for a word length of eight bits, the N for no parity and the 1 for one stop bit. The D disables the Xoff status.

Once you set the communication status on the Model 100, it's held in memory until you switch it to something else. Also set duplex on the Model 100 at full (which becomes the default). If you're communicating by phone, set the C-64's modem to originate and the 100's to answer, regardless of who makes the call.

COMMUNICATING

After you've set up the machines, load and run the 100 & 64 program on the C-64 and select the modem program on the Model 100. You're now ready to communicate between machines by typing in your message. All direct correspondence is printed on the screen of each computer. My program doesn't include a blinking cursor, but this shouldn't be much of a problem.

In addition to direct correspondence, a file can be sent from the Model 100 to the Commodore, then saved on disk. Pressing F1 on the 64 clears the screen and displays the following options:

1. Return to Terminal mode
2. Download screen code file
3. Download CBM ASCII file
4. Download Basic file
5. Quit

Selecting option 1 returns you to direct communication, while option 5 terminates the program. Options 2, 3 and 4 let you transfer one of the three types of files. When you select one of these options, the screen on the 64 clears, and a Ready message is displayed. A message also appears on the Model 100 to indicate the type of file the C-64 is expecting.

To transfer files from the Model 100, first press the F3 key for the Upload mode, as displayed on the label line. Then type in the name of the file to transfer. When the program asks for the width, press the return key. Setting the width is important for transferring to some devices, but no setting is necessary for the C-64.

Transfer now begins and is indicated on the Model 100 by the Up label changing to reverse

video. When the Up label changes back to normal, the transfer is complete.

When a Basic file is transferred to the 64, the file appears on the 64's screen and also goes into memory. Screen code and CBM ASCII files are placed in RAM but are not displayed.

After the transfer, the screen on the 64 clears, and you are asked to enter a filename. Then the program saves the file to disk. Make sure you're not duplicating an existing filename, or the save won't work. When the save is done, the menu reappears.

The C-64 expects a start-and-stop flag from the Model 100 to indicate the beginning and end of the file. This flag is equal to a byte with all bits on—a CHR$(255). You enter a CHR$(255) on the Model 100 by holding down the shift and GRPH keys while pressing the C key. This produces a shaded block similar to the graphics character you get on the 64 by pressing the Commodore key and the plus sign simultaneously.

With a Basic program, you send this code manually before pressing F3 to upload the file and again after the transfer is finished. When transferring a screen code or CBM ASCII file, however, it's easier to put the flag right in the text as the first and last characters in the file. The Model 100's text editor lets you do this using the procedure described above.

SCREEN CODE EMULATION

It's easy to enter text on the Model 100 to emulate either CBM ASCII or a word processor that uses screen codes. Regular text is typed in as usual; the only change occurs when you want to enter a nonprinting command. With most CBM ASCII and screen code files, you do this with a series of keystrokes that makes the characters you type appear in reverse video on the screen.

To emulate this with the Model 100, hold down the CTRL key and press the P key twice, then enter the character you want to appear in reverse video. The character won't appear in reverse on the Model 100's screen, but it will be preceded by an up arrow (^) and a capital P. This symbol is quite easy to spot on the Model 100, and the procedure closely follows the pattern for entering commands with both CBM ASCII and screen code programs.

When the C-64 receives this pattern through the 100 & 64 program, the program converts it to the proper code for the type of word processor you specified. For example, when using Easy Script (a typical CBM ASCII word

processing program), you press F3, which appears on the screen as a reverse asterisk, before most format commands. To set the left margin, you press F3, then type lm and the value for the left margin.

To emulate this on the Model 100, hold down the CTRL key and press the P key twice, then type lm. On the screen you'll see an up arrow, a capital P and the lm. Then enter the value for the left margin, as with Easy Script. In this manner, you can enter all the commands used in either CBM ASCII or screen code word processing programs.

In addition to emulating Easy Script and most screen code programs, the 100 & 64 program can emulate any word processing program for the 64 that's compatible with either of these. Screen code files are saved as program files using screen codes with a carriage return having the value CHR$(31). CBM ASCII files are saved as sequential files using the regular Commodore ASCII codes. You define the characters for the commands that will appear in reverse on the screen by preceding them with the value CHR$(128).

Thus, you can use the Model 100 with the 100 & 64 program to emulate any word processing program that is compatible with either of these systems.

Remember that Basic programs are transferred and saved as Commodore ASCII sequential files, so you must convert these to regular tokenized Basic programs. The Tokenizer program on page 78 of the June 1986 issue of *RUN* will do the trick for you.

If you need a portable, battery-operated Commodore, I think you'll find the Radio Shack Model 100 one of the most useful peripherals for your computer system.■

Datafile 3.6

By Mike Konshak

RUN It Right

C-64; disk drive; printer

Datafile 3.6 is the newest version of the Datafile database-management system for the C-64—a memory-based system that uses sequential files.

Datafile, the main program of a three-part package, will not alone allow printing of records, but only writing to disk. DFPrint and DFCalc, the two accessory programs that print out Datafile 3.6 files, will be published in the March and April 1987 issues of *RUN* and in the March-April ReRUN.

With Datafile, you can create your own database, choosing the number and length of fields, as well as their titles. After you've created a record file and entered your data, the program will search, sort, delete and modify the records.

As you work with Datafile, keep in mind that it will accept only non-shifted characters when you're inputting data, and that commas, colons, semicolons and quotation marks are not allowed.

If you're adding a large number of records to a file at one sitting, be sure to save the file often, just in case the power goes out unexpectedly.

DATAFILE INSTRUCTIONS

Datafile 3.6 uses an ML fast sort routine and the DOS 5.1 wedge. First, save DATAFILE 3.6, SORT GENERATOR and INSTALL DOS 5.1 to a newly formatted disk. Next, run SORT GENERATOR. When run, this creates and saves the ML sort file to disk. Lastly, run INSTALL DOS5.1. This lets you copy DOS5.1 from your Test Demo disk onto your Datafile disk.

After doing this, you are ready to use Datafile. Just type LOAD "DATAFILE",8 and RUN. The two utilities will load and run, and the main menu will appear as follows:

```
CREATE NEW FILE
QUIT PROGRAM
ADD RECORD TO CURRENT FILE
```

MODIFY RECORD IN CURRENT FILE
DELETE RECORD IN CURRENT FILE
VIEW OR EDIT FILE
SORT RECORDS BY FIELD
PRINT RECORDS USING DFPRINT/
DFCALC
READ (LOAD) OLD FILE FROM DISK
WRITE (SAVE) CURRENT FILE
TO DISK
@ DISK DRIVE COMMANDS
$ 4 DIRECTORY

You choose a menu option by pressing the key for the first letter of the option. If your program ever crashes or locks up because of a disk drive or printer error, just type GO 68 <return> to get back to the main menu without losing any record data.

CREATING A NEW RECORD FILE

Now you need to create a datafile, so enter C for the Create option. When you create a record file, you're defining the structure to which all its records must conform, so evaluate carefully the needs of your application. It's rather difficult to change your mind later, although it's possible by using Datafile utility programs. (See *RUN*, November 1985, for the DFRestructure utility.) The rules for creating record file structures are as follows:

1. It's advisable to have no more than 15 fields.
2. As indicated above, field titles cannot contain quotation marks, commas, colons or semicolons.
3. Field length, including the field title, cannot exceed 80 characters.

Now let's create a sample record file for keeping track of club members. The file MEMBERS will have the following structure:

FIELD	TITLE	LENGTH
1	LAST NAME	15
2	FIRST NAME	15
3	STREET	30
4	CITY ST	22
5	ZIP	7
6	PHONE	12
7	DATE JOINED	8

Enter this information and then watch for the display that tells you how many records the structure can hold. If the structure is a good size and otherwise satisfactory, press A to accept it. We'll assume this sample structure is all right. You now have a current file in memory.

MODIFYING RECORDS

Now, type about ten records into your file for some data to experiment with. When you press M, for modify, on the main menu, Datafile will ask which records you want to change. If you want to change just one, and you know its number, type the number and press return. If you don't know the number or you want to change a number of records,

press A to view all the records in the file, one at a time.

The current data in each field in each record will be displayed in turn. If you want to leave the field as is, press return. If you want to erase its data, press the > key. (The > will stay within that record until the file has been saved and reloaded.) You can also copy a field's data by pressing the equals key. At the end of each record, press N to advance to the next record or E to exit.

Unless you know the record number or have changes to make on every record, it's more convenient to use the alternative method of modifying records available through the View or Edit option on the main menu.

DELETING RECORDS

When you press D for deleting records, you once again have to designate a record or press A to go through the whole file. If you can't remember the record number, go to the View or Edit option on the main menu and delete the record from there.

Before Datafile will delete a record, it displays the record's entire contents on the screen. If you're sure you want to delete it, press shift/D.

As the deletion occurs, the count of records in the file decreases, and all the records after the deleted one are renumbered accordingly. To put all your records back into order, you have to use the Sort option in the main menu. Remember to save your revised file to disk.

VIEW OR EDIT FILE

This option offers the most flexibility for viewing, scanning and editing the current file. As each record is displayed, you'll be given the following eight choices:

NEXT	LAST	JUMP	FIND
MODIFY	DELETE	PRINT	EXIT

Next makes the screen step to the next record, Last steps it backward to the previous record, and Jump takes it directly to a particular record number, instead of stepping one by one. Print sends the record currently on the screen to your printer.

Find lets you locate records having common data within a certain field. Then you can modify or delete each record. When you're using Find, the screen displays a list of the field names in your current datafile and asks you to enter the number of the one you wish to search. The field name is then displayed, and you must enter the common item. Type in the string of text you're looking for and press return.

For example, if you choose a first-name field, you might enter the string JIM. The computer

would search out all the records that begin with JIM in the first-name field. Not only would JIM come up, but also JIMMY, because it begins with JIM.

SORTING

When you pick the Sort option at the main menu, the screen displays the names and numbers of the fields in the file that's in memory. You can sort the file by up to five fields, all in ascending order, and the sort will take less than 10 seconds.

Datafile stores all data as strings, not as actual numbers. For this reason, the value of each field, when compared for sorting, is determined by the position of each character. Therefore, be sure to be consistent with the format when you're entering field data.

WRITING FILES TO DISK

To save (write) a file to disk, choose W at the main menu. Datafile will ask for the name of the file, and save the file after you respond. The name may be up to 12 characters long.

When you save a record file, Datafile automatically attaches the four-character prefix DF] <space> to the filename. For instance, the name of your sample file will become DF] MEMBERS. This prefix will show up when you list the directory of the files, but you usually won't need to use the prefix yourself.

Any time a record file is written onto a disk where a file with the same name resides, Datafile makes the earlier version a backup and assigns it the suffix .BAK. Therefore, when your sample file is saved the second time, the first version will be retained on the disk with the name DF] MEMBERS.BAK.

You can load the earlier version from the Read Old File option on the main menu. To do so, enter only your filename with the suffix—MEMBERS.BAK here. Don't include the prefix that shows in the directory.

Datafile keeps only one generation of backups, so the third time you save MEMBERS, the first version will disappear. If, for some reason, you want to keep more than one generation of backups, you must give the older ones a different filename.

READING FILES FROM DISK

You'll usually pick the Read option from the main menu at the start of a Datafile session. It loads a file you've saved previously. After you've entered R, the program displays all the available files and asks which one you want to load. Type in its name and press return. The file will load, and Datafile will return to the main menu. The program will

also return to the main menu if you press the return key without typing a filename.

Remember, don't type in the four-character prefix when you enter a filename to be read; just type the name to the right of the bracket and space.

DISK COMMANDS

Datafile has five disk commands. You access the disk command menu, which contains these options plus another that returns you to the main menu, by pressing the @ key at the main menu. The disk-command options are as follows:

FORMAT

This feature allows you to format a blank disk to use for saving files. Insert the disk into the drive, then enter a disk name (up to 16 characters long), a comma, and a 2-character disk ID (any combination of numbers and letters)—for example DATAFILE FILES,D2. Follow this sequence with a return. The drive will whir for about 3½ minutes while it's formatting the disk, then return you to the main menu.

Make sure the disk you place in the drive for formatting is really the one you want to use, because this process will erase the entire disk!

DISK DIRECTORY

To list the directory of the disk currently in the drive, press the 4 key. After you've finished viewing the directory, press any key to return to the disk menu.

SCRATCH A FILE

To scratch any sequential file on the disk, enter the filename, including the DF]<space> prefix, at the prompt and press return. For instance, to scratch your sample file, you'd type DF] MEMBERS. Be sure to type in the name *exactly* as it appears in the directory, so you don't scratch the wrong file by mistake.

RENAME A FILE

To rename a sequential file, enter the old name exactly as shown in the directory, then the new name when the prompt appears. Be sure to include the special prefixed characters; otherwise, Datafile won't recognize the newly named file, and you won't be able to load it from the main menu.

VALIDATE A DISK

This option removes any corrupted files (splat files, with an * beside them in the directory) from your disk.

Again, we remind you that the programs for printing your records, DFPrint and DFCalc, will appear in *RUN's* March and April 1987 issues and in ReRUN for those months.■

The Functional Computer

By Jerald A. Brown

RUN It Right

C-64; C-128; C-16; VIC-20 (any memory size); Plus/4

Are you disappointed that Commodore Basic doesn't provide as many math functions as some calculators? Thirty-two Math Functions is a program I designed to fill this void for students, engineers and anyone else who wishes their computer offered more scientific functions. Those who need additional trig capability to design graphics programs will find it especially useful.

The program consists of a set of definitions that you can easily add to any Basic 2.0 or 7.0 program using the DEF FN command. It will run on any Commodore computer, it's short and easy to type in, and, best of all, you can use the new functions in either Immediate or Program mode.

Table 1 is a list of the functions I've included in the program. You'll find an English definition of each function and the abbreviation the program uses for it.

PROGRAM DESIGN

I deliberately established several constraints in developing the program, to make it as independent and flexible as possible. First, although I could have added other capabilities through subroutines, I wanted to use only the DEF FN command so the program could be memory resident in Immediate mode.

Second, I could have reduced many of the formulas to one conversion number and represented the value of pi by the pi sign, but I deliberately left them in their equation form for learning purposes. For example, 180/3.141592654, which is used several times, could be simplified to 57.29577951. However, leaving it as is will remind—or in-

Table 1. The 32 functions.

Common logs (base 10)	FN CL(X)
Common antilogs (base 10)	FN AL(X)
Sine in degrees	FN SN(X)
Cosine in degrees	FN CS(X)
Tangent in degrees	FN TN(X)
Arcsine in degrees	FN IS(X)
Arccosine in degrees	FN IC(X)
Arctangent in degrees	FN IT(X)
Arcsine in radians	FN AS(X)
Arccosine in radians	FN AC(X)
Sine in grads	FN SI(X)
Cosine in grads	FN CO(X)
Tangent in grads	FN TA(X)
Arcsine in grads	FN RS(X)
Arccosine in grads	FN RC(X)
Arctangent in grads	FN RT(X)
Convert degrees to radians	FN DR(X)
Convert radians to degrees	FN RD(X)
Convert degrees to grads	FN DG(X)
Convert grads to degrees	FN GD(X)
Convert radians to grads	FN RG(X)
Convert grads to radians	FN GR(X)
Convert Celsius to Fahrenheit	FN FD(X)
Convert Fahrenheit to Celsius	FN CD(X)
Convert millimeters to inches	FN IN(X)
Convert inches to millimeters	FN MM(X)
Convert liters to gallons	FN GA(X)
Convert gallons to liters	FN LI(X)
Convert kilograms to pounds	FN LB(X)
Convert pounds to kilograms	FN KG(X)
Convert grams to ounces	FN OZ(X)
Convert ounces to grams	FN GR(X)

form?—you that 180 degrees is one-half of a circle, 3.141592654 is the value of pi, and the first divided by the second gives the number of degrees in one radian, a radian being the unit of measure of an angle at the center of a circle whose intercepted arc equals the circle's radius. Designing the functions this way displays their logical basis and helps the user learn.

My final programming constraint was to make the functions stand alone; that is, none of them reference previously defined functions. This way, you can type in only as many as you want, and you can execute any function or group of functions without referencing the others.

USING THE PROGRAM

Load and run MATH FUNCTIONS. To use a function in Immediate mode, type PRINT and the function abbreviation in the Table, followed by a return. For example, to calculate the sine of 30 degrees, type PRINT FN SN(30) <return>. The value .5, which is the sine of 30 degrees, will appear on the screen.

To use a function in a program of your own, you must reference it with a DEF FN statement. Be sure to place these statements at the beginning of your program listing, so the functions are in place before the program needs them.

Keep in mind that you can use the functions only if they're in memory. If you enter a New command, your computer will lose its added mathematical capabilities.

MORE FUNCTIONS?

Commodore has provided even more complex formulas in their computer manuals. See the *C-64 Programmer's Reference Guide*, the *C-128 System Guide* and the *VIC-20 Programmer's Reference Guide* for charts showing a number of other scientific functions you might find useful. You might also want to consult your manual to learn how to enter numbers in scientific notation.■

Monthly Labels

By John Hundley

RUN It Right

C-64; printer

One of the tasks I wanted to use my printer for, other than word processing, was to make mailing labels for my bills. I soon realized, however, that although there were several good label-making programs on the market and in the public domain, none of them fitted my needs. I could print out labels for a whole mailing list, but when I wanted a year's supply for just one name, I had to print one label and answer the prompt, "print another list?" 12 times.

After you load the program, run it, making sure your printer is on. After a couple of seconds, a menu will appear with a choice of several printers. Press the number next to the type of printer you are using.

Next, type in the name of the person or business you are mailing to. After you press the return key, you'll be prompted for the address. Although the city and state appear here on separate screens, they'll appear on the same line on the label.

The next prompt will ask for the number of line spaces between labels. You'll have to experiment to get the proper spacing. I use a Star SG-10 printer and input 11 spaces for a one-inch label. The Commodore printers won't need nearly that many spaces; in fact, you might start out trying only one or two.

The last prompt is for the number of copies you want. When the printing is finished, a bell will ring.

If your printer is not one of those I've included on the menu, try experimenting with others listed, or choose number five for "other." If you do choose "other," you may have to try different label widths to get the proper horizontal spacing. Monthly Labels should work with most printers.■

Lots of Labels

By Chris Achtschin

RUN It Right

C-64; printer

When I purchased my Epson printer, one of the first projects I wanted to do was print some return address labels. However, I discovered that the mailing list programs I had wouldn't repeat one particular label a number of times. So, I wrote Label Copy to fill the gap. I've run it on a Commodore 1525 printer as well as on my Epson RX-80 F/T, and it should run on any C-64-compatible printer.

Label Copy will print up to five lines on a label for a specified number of prints. You can also determine the number of vertical spaces between the last line and the new first line, thus adjusting to the depth of the labels you're using. Printing some test labels will get the text alignment right.

After you've entered your text, the program will display it so you can check it for errors. If you say everything is okay, the labels will begin printing with the current label number displayed on the screen.

You can save your label text on disk if you intend to reuse it. When the prompt appears for a "save name," use a short one that will remind you of the text. You can use a save name only once.

Label Copy includes defaults for some of the text parameters. To customize the program to your own needs, you can easily change these defaults by changing the poked characters. For example, line 50 pokes screen location 1078 with 14 if you type N for no instructions. A 25 would mean Y for yes. You can find the screen display codes in the *Commodore C-64 User's Guide.*

Label Copy could be used for much more than return address labels. Some other applications might include name tags, owner tags for luggage, books or record albums and ID tags for appliances.■

Envelope Maker

By Michael Broussard

RUN It Right

C-64; printer

If your handwriting is as sloppy as mine, you probably write letters with your word processor. However, when it's time to mail the letter, chances are you end up addressing the envelope by hand, because it's too much trouble to get out the typewriter or load labels into the printer. Don't your letters deserve to be as neat on the outside as they are on the inside? Here's a utility that can help.

Envelope Maker is a Basic program that uses your printer to create envelopes in a couple of common sizes, with the address printed right on the front. It takes regular 8½-by-11 paper, and you can choose whatever color you like. After the printer is done, you cut on the dotted lines, fold and fasten with tape or a bit of glue. The procedure is fast and easy, the envelopes are professional looking and your postman will love them.

To use Envelope Maker, you will have to customize it. List the Data statements in lines 780 through 810, then type in your own return address in place of mine, placing each address line in quotes. The program expects four lines of return address, but I used only three, so the last Data statement specifies an empty string.

FORMATTING ENVELOPES

When you run Envelope Maker, the screen clears, and it asks which kind of envelope you want, standard letter or French fold. Standard letter is $6\frac{7}{16}$-by-$3\frac{13}{16}$ inches, a size often used for informal correspondence or paying bills. French fold is a greeting-card-type envelope of 5¾-by-4½ inches. French fold is the perfect size for a card produced by Broderbund's Print Shop. If you've been searching in vain at stationery stores for plain white envelopes to use with Print Shop cards, look no further! Choose the envelope type by pressing 1 or 2.

Next, the program asks if you want to use the default return address. If you respond Y, for yes, Envelope Maker automatically uses the return address specified in the Data statements at the end of the program. If you answer N, it asks for up to four lines of return address, one line at a time. To skip part of an address, press the return key when the prompt for that line appears.

Now you must enter the address of the person to whom you are sending the letter. Type in the address lines, one at a time, pressing return after each. Again, you may specify up to four lines.

Then you are asked if you want to center the address lines. If you enter Y, the lines will be centered in relation to each other. If you specify N, the lines will align on the left, but the block will still be centered on the envelope. In either case, the address is double spaced.

CREATING AN ENVELOPE

With the formatting done, make sure the paper in your printer is set at the top of a new page and press any key to print the envelope. When output is complete, Envelope Maker will ask if you want to make another. If you answer Y, the program will loop back to prompt for more address lines. If you answer N, the program will end. Note that if you answer Y, the program assumes you want the same size envelope with the same return address. If you want to change either of these parameters, answer N and run the program again.

To assemble the envelope, cut along the outside lines, then fold along the inside lines. First fold in the little side flaps, then fold up the bottom. Fasten with glue or tape (I use a glue stick, available at most office supply stores). Finally, insert your letter and fold and fasten the main flap.

Envelope Maker is designed to run on printers with pica type (10 characters per inch). If you have a printer with elite type (12 characters per inch), such as Commodore's 1526, change the value assigned to the variable ELITE on line 100 of the program from 0 to 1. Also, Envelope Maker assumes that your printer is device 4. If you're using a different device number for your printer, substitute the appropriate value in the variable PDEV in line 100.

The printer is opened with a device subaddress of 7, so the envelope is printed with the upper/lowercase character set. If you prefer the uppercase/graphics character set, change the value assigned to the variable SA on line 100 from 7 to 0.

CUSTOMIZED INPUT ROUTINE

As an extra benefit, Envelope Maker contains a short machine language subroutine you can include in your own programs. This routine functions as an alternative to Basic's standard Input statement and behaves like the Input statement with the following exceptions:

1. It doesn't print a ? prompt, so you can specify whatever prompt you like.

2. It accepts the whole line you type as input, even if the line contains commas or colons.

3. Because of the second exception, it inputs a value for only one string variable at a time.

To use the subroutine, your program must first poke it into memory. Envelope Maker does this in lines 120 and 130. You should include these lines in your own program, along with the Data statements numbered 720 through 750.

Envelope Maker pokes the subroutine into RAM starting at location 49152, but you can move it anywhere you like by changing the value assigned to the variable SUB on line 120. (Just be sure you don't put it where it will interfere with Basic.) The subroutine is invoked with a SYS statement that also specifies the name of the variable to be input. Here's how a typical call might look:

```
PRINT"Input your name: ";:SYS
  SUB,N$:PRINT
```

Assuming the subroutine has already been poked in beginning at the RAM location indicated by SUB, this line first prints a prompt and then assigns a string to the variable N$.■

Master Menus

By Jim Pellechi

RUN It Right

C-64; disk drive

One of the most important elements of any successful program is its introduction. The appearance of the opening displays can win or lose an audience.

The two Master Menus programs provide a professional-looking main-menu screen with colorful, animated special effects. You can easily adapt the menu to any program you might develop.

The two programs consist of the Basic main program, entitled MENU DEMO, and a Basic loader called ML MENU LOADER that creates windows. These two programs are designed to go at the beginning of your program listing.

CUSTOMIZING THE MENU

By making slight changes to a few lines in the MENU DEMO program, you can design the menu for your program's needs. I set it up for a maximum of seven menu choices, to be selected with keys 1–7. If you need fewer choices in your program, alter line 1120 to check for fewer keypresses. For example, to test for keys 1–3, change the second CHR$ number to CHR$(51). Altering this line won't affect any other part of the program, so you won't have to rewrite portions of the listing.

In line 1250, insert a 34-character message—perhaps instructions for choosing from the menu—that will scroll continuously across the screen.

In lines 1295–1325, insert up to seven menu items, each up to 16 characters long, on the appropriate lines. The periods in the lines indicate where to enter the characters. Don't erase any periods, even if you don't use them all.

Lines 1365–1455 hold the instructions for branching to sections of your program that the user chooses. Change these locations to match your program.

Run ML MENU LOADER, then load and run MENU DEMO. You must have ML/MENU LOADER on disk with any program that uses MENU DEMO.■

Please send me back issues of ReRUN

____ Fall Edition
____ Winter Edition
____ January/February 1986 **
____ March/April 1986
____ May/June 1986
____ July/August 1986
____ Productivity Pak II
____ September/October 1986
____ November/December 1986

____ Cassette version(s) at $11.47*
____ Disk version(s) at $21.47*

* *Prices include postage and handling. For foreign air mail, please add U.S. $1.50 per item and $25 per subscription. Prepayment only.*
** *All 1986 editions contain 128-mode programs and are available on disk only.*

☐ Payment Enclosed ☐ MC ☐ VISA ☐ AE

Card # Exp. Date

Name

Address

City State Zip

Signature

ReRUN • 80 Elm Street • Peterborough, NH • 03458

BEAT THE RUSH!

Please send me:

☐ **1 year (6 issues) for $89.97**
☐ **March/April 1987 ReRUN disk for $21.47.***

Available in April 1987.

Includes programs for C-64 and C-128 (in both 64 and 128 modes).

Price includes postage and handling. For foreign air mail, please add U.S. $1.50 per item and $25 per subscription. Prepayment only.

☐ Payment Enclosed ☐ MC ☐ VISA ☐ AE

Card # Exp. Date

Name

Address

City State Zip

Signature

ReRUN • 80 Elm Street • Peterborough, NH • 03458

23 *RUN* Programs Included on this Disk:

Calendar ▸ Database Management ▸ Finance ▸
Games ▸ Communications ▸ Education ▸
Programming Aids ▸ Print Utility ▸
Easy Applications ▸ 1986 RUN Index

From the January RUN:

- Reminder 128
- RUN Script 128, Part 2
- 64 Notepad Update
- Word Wars
- Solving the Split-Word Problem
- Mega-Magic (Short Sorter)
- Easy Applications (Keeping Up to Date)
- Resource Center (Spelling)

From the February RUN:

- Datafile 3.6
- Patching Up RUN Script 128
- Hook Up to a Portable
- Retir'eze
- Twiddle
- The Functional Computer
- Break the 128 Memory Barrier!
- Mega-Magic (C 128 Hi Res Screen Dump)
- Easy Applications (Monthly Labels)

PLUS! From the RUN Special Issue #3:

- Mind Your Mortgage
- The Light Choice
- Add/Calc 128
- Lots of Labels
- Envelope Maker
- Master Menus
- Index of 1986 RUN Articles

If any manufacturing defect becomes apparent, the defective disk will be replaced free of charge if returned by prepaid mail within 30 days of purchase. Send it, with a letter specifying the defect, to:

ReRUN • 80 Elm Street • Peterborough, NH 03458

Replacements will not be made if the disk has been altered, repaired or misused through negligence, or if it shows signs of excessive wear or is damaged by equipment.

The programs in ReRUN are taken directly from listings prepared to accompany articles in *RUN* magazine. They will not run under all system configurations. Use the RUN It Right information included with each article as your guide.

The entire contents are copyrighted 1986 by CW Communications/Peterborough. Unauthorized duplication is a violation of applicable laws.

©Copyright 1987 CW Communications/Peterborough